Proceedings of the 2005 IEEE
International Conference on Robotics and Automation
Barcelona, Spain, April 2005

Robust Object Detection at Regions of Interest with an Application in Ball Recognition

Sara Mitri, Simone Frintrop,
Kai Pervölz, Hartmut Surmann
Fraunhofer Institute for Autonomous Intelligent Systems (AIS)
Schloss Birlinghoven,
D-53754 Sankt Augustin, Germany
simone.frintrop@ais.fraunhofer.de

Andreas Nüchter
University of Osnabrück
Institute for Computer Science
Knowledge-Based Systems Research Group
Albrechtstraße 28
D-49069 Osnabrück, Germany

***Abstract*— In this paper, we present a new combination of a biologically inspired attention system (VOCUS – Visual Object detection with a CompUtational attention System) with a robust object detection method. As an application, we built a reliable system for ball recognition in the RoboCup context. Firstly, VOCUS finds regions of interest generating a hypothesis for possible locations of the ball. Secondly, a fast classifier verifies the hypothesis by detecting balls at regions of interest. The combination of both approaches makes the system highly robust and eliminates false detections. Furthermore, the system is quickly adaptable to balls in different scenarios: The complex classifier is universally applicable to balls in every context and the attention system improves the performance by learning scenario-specific features quickly from only a few training examples.**

***Index Terms*— visual attention, object classification.**

I. Introduction

A fundamental problem in the field of robotics is the perception of the environment. Our work is inspired by the biological two stage process of searching for an object in a visual scene [17]: First, human attention is caught by regions with object-specific features such as color or orientations. Second, recognition processes restricted to these regions verify or falsify these hypotheses. Our system is designed after these two stages.

This paper proposes a scheme for learning and detecting soccer balls through the combination of the computational attention system VOCUS with a classifier. Recognizing soccer balls as an application in the Robot World Cup Soccer Games and Conferences (RoboCup) [8] has been a tough problem to solve because of the lack of definite characteristics describing a ball. Our solution is reliable, scale-independent and color-adaptable in the sense that it can be applied to balls of any size, surface pattern and color.

Our approach consists of a training phase, an adaptation phase, and a detection phase. In the training phase, the classifier is exhaustively trained using balls of different sizes, colors, and surface patterns from a wide variety of training images. The output of the training is a cascade of classifiers that in turn consist of a set of decision trees. In the adaptation phase, VOCUS is quickly adapted to a special scenario: it learns from few example images (here: 2) the properties of the scenario, e.g., the color of the ball and its intensity contrast to the environment. This adaptation results in a set of feature weights describing the ball in its surroundings. In the detection phase, first,

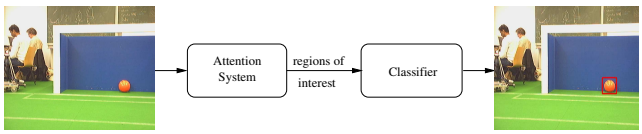

Fig. 1. The recognition system consists of the attention system VOCUS providing object candidates and a classification system verifying the hypothesis. The combination yields a flexible and robust system.

VOCUS computes regions of interest by weighting the image features with the learned weights. Second, the classifier is applied to these regions, verifying the object hypothesis (Fig. 1). This approach makes the system flexible as well as robust.

The visual attention system VOCUS consists of a bottom-up part computing data-driven saliency and a top-down part and enabling goal-directed search. Bottom-up saliency results from uniqueness of features, e.g., a black sheep among white ones, whereas top-down saliency uses features that belong to a specified target, e.g., red when searching for a red ball. The bottom-up part, also described in [7], is based on the well-known model of visual attention by Koch & Ullman [11] used by many computational attention systems [12], [1]. It computes saliencies according to the features intensity, orientation, and color and combines them in a saliency map. The most salient region in this map yields the focus of attention. The top-down part is new: it uses previously learned feature weights to excite target-specific features and inhibit others.

Balls are classified according to the Viola-Jones classifier [22]: The shape of the ball is learned by using edge-filtered and thresholded images, represented by computationally efficient integral images [22]. The Gentle Ada Boost learning technique [5] is used to learn a selection of Classification and Regression Trees (CARTs) that select an arrangement of Haar-like features to classify the object. Several selections are combined into a cascade of classifiers. This learning phase is relatively time-consuming, but only needs to be executed once, since the classifier is then general enough to apply to any ball shaped object.

The most common techniques for ball detection in RoboCup rely on color information. In the last few years, fast color segmentation algorithms have been developed to detect and track objects in this scenario [10], [19]. The community agreed that in the near future, visual cues like color coding will be removed to come to a more realistic setup with robots playing with a “normal” soccer ball [20].

0-7803-8914-X/05/$20.00 ©2005 IEEE.

Treptow and Zell learn with Ada Boost conglomerations of Haar like classifiers and arrange them in a cascade to recognize balls without color information [20]. However, in previous work [16] we show problems with learning non symmetric object patterns in differently illuminated environments. To overcome this problem, we preporcessed the input with edge detection and learned classification and regression trees (CARTs) instead of simple conglomerations of feature classifiers and accomplished color-independent ball detection for various balls. To reduce a significant amount of false detections, where the classifier marked various round shapes, e.g., the heads in Fig. 7, we propose here an attention algorithm that is quickly adapted on the spot to a specific ball. It yields several region hypotheses. With the combination of both systems, we eliminate the false detections and identify only the intersection of the two classified sets as correct. In this way, the ball detector can efficiently be applied to more complex images, without worrying about false detections.

The combination of an attention system with classification has also been done by Miau, Papageorgiou and Itti who detect pedestrians on attentionally focused image regions using a support vector machine algorithm [15]. Walther and colleagues combine in [23] an attention system with the object recognizer of Lowe [14] and show that the recognition results are improved by the attentional front-end. Nevertheless, all of these approaches focus on bottom-up attention and do not enable goal-directed search. To our knowledge, this is the first approach combining a top-down modulated attention system with a classifier.

The rest of the paper is structured as follows: First, we describe the attention system VOCUS in section II. We then discuss briefly the process of learning and detecting balls in section III. The results of each algorithm independently as well as in combination are given in section IV and, finally, section V concludes the paper.

II. The Attention System VOCUS

In this section, we present the goal-directed visual attention system VOCUS (Visual Object detection with a CompUtational attention System) (cf. Fig. 2). With visual attention we mean a selective search-optimization mechanism that tunes the visual processing machinery to approach an optimal configuration [21]. VOCUS consists of a bottom-up part computing data-driven saliency and a top-down part enabling goal-directed search. The global saliency is determined from bottom-up and top-down cues. In the following, we first describe the computation of the bottom-up and then of the top-down saliency.

A. Bottom-up saliency

1) **Feature Computations:** The first step for computing bottom-up saliency is to generate image pyramids for each feature to enable computations on different scales. Three features are considered: Intensity, orientation, and color. For the feature intensity, we convert the input image into gray-scale and generate a Gaussian pyramid with 5 scales s_0 to s_4 by successively low-pass filtering and subsampling

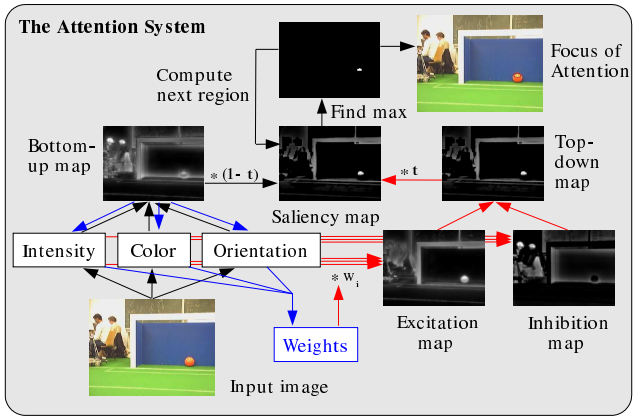

Fig. 2. The goal-directed visual attention system VOCUS with a bottom-up part (left) and a top-down part (right). In learn mode, target weights are learned (blue line arrows). These are used in search mode (red short arrows).

the input image, i.e., scale $(i+1)$ has half the width and height of scale i.

The intensity maps are created by center-surround mechanisms, which compute the intensity differences between image regions and their surroundings. We compute two kinds of maps, the on-center maps I''_{on} for bright regions on dark background, and the off-center maps I''_{off}: Each pixel in these maps is computed by the difference between a center c and a surround σ (I''_{on}) or vice versa (I''_{off}). Here, c is a pixel in one of the scales s_2 to s_4, σ is the average of the surrounding pixels for two different radii. This yields 12 intensity scale maps $I''_{i,s,\sigma}$ with $i \in \{\text{on, off}\}$, $s \in \{s_2\text{-}s_4\}$, and $\sigma \in \{3, 7\}$.

The maps for each i are summed up by *inter-scale addition* $\bigoplus$, i.e., all maps are resized to scale 2 and then added up pixel by pixel yielding the intensity feature maps $I'_i = \bigoplus_{s,\sigma} I''_{i,s,\sigma}$.

To obtain the orientation maps, four oriented Gabor pyramids are created, detecting bar-like features of the orientations $\theta = \{0°, 45°, 90°, 135°\}$. The maps 2 to 4 of each pyramid are summed up by inter-scale addition yielding 4 orientation feature maps O'_θ.

To compute the color feature maps, the color image is first converted into the uniform CIE LAB color space [2]. It represents colors similar to human perception. The three parameters in the model represent the luminance of the color (L), its position between red and green (A) and its position between yellow and blue (B). From the LAB image, a color image pyramid P_{LAB} is generated, from which four color pyramids P_R, P_G, P_B, and P_Y are computed for the colors red, green, blue, and yellow. The maps of these pyramids show to which degree a color is represented in an image, i.e., the maps in P_R show the brightest values at red regions and the darkest values at green regions. Luminance is already considered in the intensity maps, so we ignore this channel here. The pixel value $P_{R,s}(x,y)$ in map s of pyramid P_R is obtained by the distance between the corresponding pixel $P_{\text{LAB}}(x,y)$ and the prototype for red $r = (r_a, r_b) = (255, 127)$. Since $P_{\text{LAB}}(x,y)$ is of the form (p_a, p_b), this yields: $P_{R,s}(x,y) = ||(p_a,p_b),(r_a,r_b)|| = \sqrt{(p_a - r_a)^2 + (p_b - r_b)^2}$.

On these pyramids, the color contrast is computed by on-center-off-surround differences yielding 24 color scale maps $C''_{\gamma,s,\sigma}$ with $\gamma \in \{\text{red, green, blue, yellow}\}$, $s \in \{s_2\text{-}s_4\}$, and $\sigma \in \{3, 7\}$. The maps of each color are inter-scale added into 4 color feature maps $C'_\gamma = \bigoplus_{s,\sigma} \hat{C}_{\gamma,s,\sigma}$.

2) **Fusing Saliencies:** All feature maps of one feature are combined into a conspicuity map yielding one map for each feature: $I = \sum_i \mathcal{W}(I'_i)$, $O = \sum_\theta \mathcal{W}(O'_\theta)$, $C = \sum_\gamma \mathcal{W}(C'_\gamma)$. The bottom-up saliency map S_{bu} is finally determined by fusing the conspicuity maps: $S_{bu} = \mathcal{W}(I) + \mathcal{W}(O) + \mathcal{W}(C)$

The exclusivity weighting $\mathcal{W}$ is a very important strategy since it enables the increase of the impact of relevant maps. Otherwise, a region peaking out in a single feature would be lost in the bulk of maps and no pop-out would be possible. In our context, important maps are those that have few highly salient peaks. For weighting maps according to the number of peaks, each map M is divided by the square root of the number of local maxima m that exceed a threshold t: $\mathcal{W}(M) = M/\sqrt{m} \quad \forall m : m > t$. Furthermore, the maps are normalized after summation relative to the largest value within the summed maps. This yields advantages over the normalization relative to a fixed value (details in [7]).

3) **The Focus of Attention (FOA):** To determine the most salient location in S_{bu}, the point of maximal activation is located. Starting from this point, region growing recursively finds all neighbors with similar values within a threshold and the FOA is directed to this region. Finally, the salient region is inhibited in the saliency map by zeroing, enabling the computation of the next FOA.

B. Top-down saliency

1) **Learning mode:** In learning mode, the user marks a rectangle in a training image specifying the region that has to be learned. Then, VOCUS computes the bottom-up saliency map and the most salient region inside the rectangle. So, the system is able to determine automatically what is important in a specified region. It concentrates on parts that are most salient and disregards the background or less salient parts.

Next, weights are determined for the feature and conspicuity maps, indicating how important a feature is in the specified region. The weights are the quotient of the mean saliency in the target region m_r and in the background $m_{(image-r)}$: $w_i = m_r/m_{(image-r)}$. This computation considers not only which features are the strongest in the region of interest, it regards also which features separate the best region from the rest of the image.

Several training images: Learning weights from one single training image usually yields good results if the target object occurs in all test images in a similar way, i.e., on a similar background. To enable a more stable recognition even on varying backgrounds, we determine the average weights from several training images by computing the geometric mean of the weights, i.e., $w_{i,(1..n)} = \sqrt[n]{\prod_{j=1}^{n} w_{i,j}}$, where n is the number of training images.

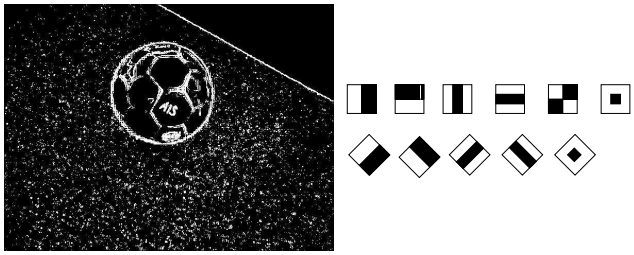

Fig. 3. Left: Sobel filter applied to a colored image and then thresholded. Right: Edge, line, diagonal, center surround and 45° features are used for classification.

An algorithm for choosing the training images is proposed in [6]. It showed that, usually, even in complex scenarios 5 training images suffice; for ball detection, already two training images yielded the best performance.

2) **Search mode:** In search mode, firstly the bottom-up saliency map is computed. Additionally, we determine a top-down saliency map that competes with the bottom-up map for saliency. The top-down map is composed of an excitation and an inhibition map. The excitation map E is the weighted sum of all feature maps that are important for the learned object, namely the features with weights greater than 1. The inhibition map I contains the feature maps that are not present in the learned object, namely the features with weights smaller than 1:

$$\begin{aligned} E &= \frac{\sum_i (w_i * \text{Map}_i)}{\sum_j (w_j)} & \forall i : w_i > 1, \\ I &= \frac{\sum_i ((1/w_i) * \text{Map}_i)}{\sum_j (w_j)} & \forall i : w_i < 1. \end{aligned}$$

The top-down saliency map $S_{(td)}$ is obtained by: $S_{(td)} = E - I$. The final saliency map S is composed as a combination of bottom-up and top-down influences. When fusing the maps, it is possible to determine the degree to which each map contributes by weighting the maps with a top-down factor $t \in [0..1]$: $S = (1-t) * S_{(bu)} + t * S_{(td)}$.

With $t = 1$, VOCUS looks only for the specified target. With $t < 1$, also bottom-up cues have an influence and may divert the focus of attention. This is also an important mechanism in human visual attention. E.g., a person suddenly entering a room catches immediately our attention, independently of the task. For the application discussed in this paper, we always use $t = 1$ and use the bottom-up saliency only to learn the weights of the training objects. Thus, the robot focuses its attention completely on the ball and not to play foul on other robots.

III. Color-Independent Ball Classification

In this section we briefly discuss the classifier for ball detection that is applied to the foci of attention. The algorithm here refers to previous work discussed in [16], which was inspired by Viola and Jones' boosted cascade of simple classifiers for fast face detection [22].

A. Color Invariance using Linear Image Filters

The problem with recognizing general shapes, such as balls, as in our particular case, is the number of possibilities in the visual appearance of a ball. A ball can take on any color and size and may have any pattern on its surface. In order to generalize the concept of a ball, the initial goal

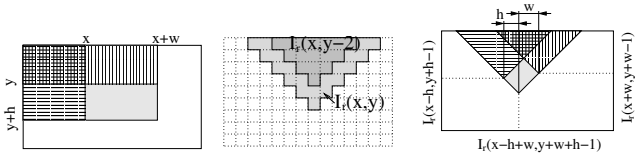

Fig. 4. Left: Computation of feature values F in the shaded region is based on the four upper rectangles. Middle: Calculation of the rotated integral image I_r. Right: Four lookups in the rotated integral image are required to compute the feature value a rotated feature F_r.

was to eliminate any color information in the data images representing the balls.

To detect the edges in the image, we use linear image filters followed by a threshold to eliminate noise data, which would then be given as input to the classifier, which in turn handles differences in size, pattern, lighting, etc. For this paper, we are using a Sobel filter, as described in [4].

In order to eliminate the color information in the images, we apply the filter to the colored image and then use a threshold t to include any pixel in any of the 3 color channels that crossed the threshold t value in the output image. The resulting image is a binary image including the thresholded pixels of the 3 color channels. A typical output image of this technique is shown in Fig. 3 (left).

This edge detection and thresholding technique is applied to all images used as input to the training of the Haar classifier. The training process is described in the following subsections.

B. Feature Detection using Integral Images

There are many motivations for using features rather than pixels directly. For mobile robots, a critical motivation is that feature based systems operate much faster than pixel based systems [22]. The features are called Haar-like, since they follow the same structure as the Haar basis, i.e., step functions introduced by Alfred Haar to define wavelets. They are also used in [13], [3], [20], [22]. Fig. 3 (right) shows the eleven basis features, i.e., edge, line, diagonal and center surround features. The base resolution of the object detector is 30×30 pixels, thus, the set of possible features in this area is very large (642592 features, see [13] for calculation details). A single feature is effectively computed on input images using integral images [22], also known as summed area tables [13]. An integral image I is an intermediate representation for the image and contains the sum of gray scale pixel values of image N with height y and width x, i.e.,

$$I(x,y) = \sum_{x'=0}^{x} \sum_{y'=0}^{y} N(x',y').$$

The integral image is computed recursively, by the formulas: $I(x,y) = I(x,y-1) + I(x-1,y) + N(x,y) - I(x-1,y-1)$ with $I(-1,y) = I(x,-1) = I(-1,-1) = 0$, therefore requiring only one scan over the input data. This intermediate representation $I(x,y)$ allows the computation of a rectangle feature value at (x,y) with height and width (h,w) using four references (see Fig. 4 (left)):

$$\begin{aligned} F(x,y,h,w) \quad = \quad & I(x,y) + I(x+w,y+h) - \\ & I(x,y+h) - I(x+w,y). \end{aligned}$$

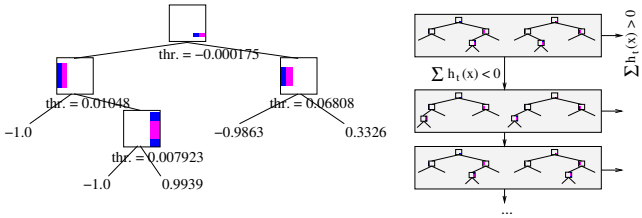

Fig. 5. Left: A Classification and Regression Tree with 4 splits. According to the specific filter applied to the image input section x, the output of the tree, $h_t(x)$ is calculated, depending on the threshold values. Right: A cascade of CARTs [16]. $h_t(x)$ is determined depending on the path through the tree.

For the computation of the rotated features, Lienhart et. al. introduced rotated summed area tables that contain the sum of the pixels of the rectangle rotated by 45° with the bottom-most corner at (x,y) and extending till the boundaries of the image (see Fig. 4 (middle and right)) [13]:

$$I_r(x,y) = \sum_{x'=0}^{x} \sum_{y'=0}^{x-|x'-y|} N(x',y').$$

The rotated integral image I_r is computed recursively, i.e., $I_r(x,y) = I_r(x-1,y-1) + I_r(x+1,y-1) - I_r(x,y-2) + N(x,y) + N(x,y-1)$ using the start values $I_r(-1,y) = I_r(x,-1) = I_r(x,-2) = I_r(-1,-1) = I_r(-1,-2) = 0$. Four table lookups are required to compute the pixel sum of any rotated rectangle with the formula:

$$\begin{aligned} F_r(x,y,h,w) = \; & I_r(x+w-h,y+w+h-1) + I_r(x,y-1) - \\ & I_r(x-h,y+h-1) - I_r(x+w,y+w-1) \end{aligned}$$

Since the features are compositions of rectangles, they are computed with several lookups and subtractions weighted with the area of the black and white rectangles.

To detect a feature, a threshold is required. This threshold is automatically determined during a fitting process, such that a minimum number of examples are misclassified. Furthermore, the return values (α, β) of the feature are determined, such that the error on the examples is minimized. The examples are given in a set of images that are classified as positive or negative samples. The set is also used in the learning phase that is briefly described next.

C. Learning Classification Functions

1) Classification and Regression Trees: For all 642592 possible features a Classification and Regression Tree (CART) is created. CART analysis is a form of binary recursive partitioning. Each node is split into two child nodes, in which case the original node is called a parent node. The term recursive refers to the fact that the binary partitioning process is applied over and over to reach a given number of splits (4 in this case). In order to find the best possible split features, all possible splits are calculated, as well as all possible return values to be used in a split node. The program seeks to maximize the average "purity" of the two child nodes using the misclassification error measure. Fig. 5 (left) shows a CART classifier.

2) Gentle Ada Boost for CARTs: The Gentle Ada Boost Algorithm [5] is used to select a set of simple CARTs to achieve a given detection and error rate [13]. In the following, a detection is referred to as a hit and an error as a false alarm.

The learning is based on N weighted training examples $(x_1, y_1), \ldots, (x_N, y_N)$, where x_i are the images and $y_i \in \{-1, 1\}, i \in \{1, \ldots, N\}$ the classified output. At the beginning of the learning phase the weights w_i are initialized with $w_i = 1/N$. The following three steps are repeated to select CARTs until a given detection rate d is reached:

1) Every classifier, i.e., a CART, is fit to the data. Hereby the error e is calculated with respect to the weights w_i.
2) The best CART h_t is chosen for the classification function. The counter t is incremented.
3) The weights are updated with $w_i := w_i \cdot e^{-y_i h_t(x_i)}$ and renormalized.

The final output of the classifier is $\text{sign}(\sum_{t=1}^{T} h_t(x)) > 0$, with $h_t(x)$ the weighted return value of the CART. Next, a cascade based on these classifiers is built.

D. The Cascade of Classifiers

The performance of a classifier is not suitable for object classification, since it produces a high hit rate, e.g., 0.999, but also a high error rate, e.g., 0.5. Nevertheless, the hit rate is much higher than the error rate. To construct an overall good classifier, several classifiers are arranged in a cascade, i.e., a degenerated decision tree. In every stage of the cascade, a decision is made whether the image contains the object or not. This computation reduces both rates. Since the hit rate is close to one, their multiplication results also in a value close to one, while the multiplication of the smaller error rates approaches zero. Furthermore, this speeds up the whole classification process.

An overall effective cascade is learned by a simple iterative method. For every stage the classification function $h_t(x)$ is learned, until the required hit rate is reached. The process continues with the next stage using the correct classified positive and the currently misclassified negative examples. The number of CARTs used in each classifier may increase with additional stages.

IV. Experiments and Results

First the performance of the classifier is shown. Then, the attention algorithm is additionally applied to adapt the detection and to reduce the false positives.

A. Results of the classifier alone

The ball detection cascade was learned with a total of 1000 images, with complex scenes included in the training set, and tested by using three soccer balls of different colors and patterns. The process of generating the cascade of classifiers is relatively time-consuming but it only needs to be executed once, provided a good cascade is generated. Fig. 6 shows detection results on five different kinds of balls, thus the CARTs form a correct dependency

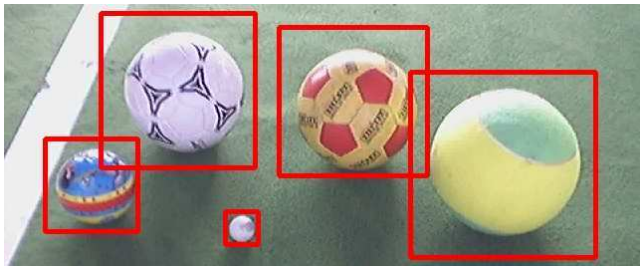

Fig. 6. Five different kind of balls are detected by the classifier.

of features. Since only the upper two balls (white and yellow/red ball) and the red one given in Fig. 7 were used for learning, the figure demonstrates the classifier's ability to generalize to all balls.

For each kind of ball we ran the test with 60 images, making a total of 180 test images. The results in Table I reveal how many red, white or yellow/red balls were correctly classified or not detected, as well as the number of false positives for each ball. The problems we were facing with this approach was the difficulty to differentiate between soccer balls and other spherical objects (Fig. 7).

TABLE I
DETECTION RATE OF THE CASCADE OF CLASSIFIER DEPENDING ON THE USED NUMBER OF STAGES. THE CASCADE WITH 10 STAGES WAS USED FOR THE EXPERIMENTS WITH THE ATTENTION SYSTEM.

	# stages	Correct	Not Detected	False Pos.
red ball		52/60	8/60	114
white ball	9	48/60	12/60	70
yel/red ball		57/60	3/60	108
Total		157/180	23/180	292
red ball		**45/60**	**15/60**	**52**
white ball	**10**	**44/60**	**16/60**	**45**
yel/red ball		**57/60**	**3/60**	**63**
Total		**146/180**	**34/180**	**160**
red ball		45/60	15/60	51
white ball	11	42/60	18/60	47
yel/red ball		56/60	4/60	65
Total		143/180	37/180	163
red ball		44/60	16/60	26
white ball	12	29/60	31/60	31
yel/red ball		37/60	23/60	23
Total		110/180	70/180	80

The detection rate of the classifier is adjustable, i.e., a lower number of stages of the cascade increases the number of detections (hits), but also the amount of false detections. By combining the classifier and the attention algorithm the false positive detection rate will be reduced.

B. Combining the classifier and the attention algorithm

The output of the combination of the two algorithms is the intersection of both result sets. The balls detected must be found both by the ball classifier as well as the attention algorithm. First, the foci are found in the image. Then, the classifier tries to detect balls at these specific regions. The results of the combination are shown in Table II. The test data is composed of a set of 60 realistic RoboCup images for each ball, where there is exactly one ball in each image. These were taken with backgrounds of different lighting (color) and complexity. The classifier searches areas of the first 5 foci found by the attention algorithm.

The combination is very useful in eliminating false positives in images. This is shown in Fig. 7, where the false

TABLE II
DETECTION RATE OF COMBINED ALGORITHM. COLUMN 2 (ATTENTION) SHOWS WHICH OF THE 5 FOCI POINTS TO THE BALL (AVERAGE).

	Att.	Classifier only		Att. and Class.	
		Found	False Pos.	Found	False Pos.
red Ball	1.0	45/60	52	45/60	3
white ball	1.0	44/60	45	41/60	0
yel/red b.	1.2	57/60	63	55/60	20
Total	1.07	146/180	160	141/180	23

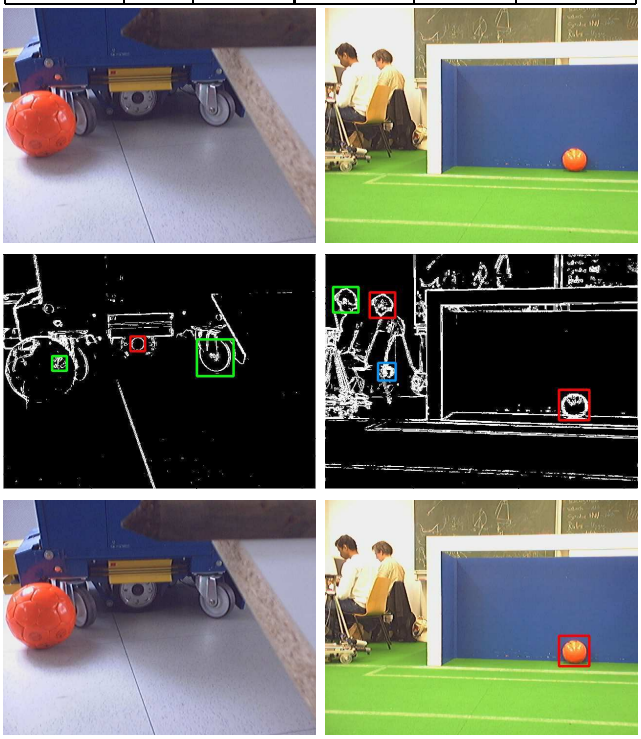

Fig. 7. Top: Input images including round objects. Middle: False alarms in filtered images. Bottom left: False positives eliminated, ball not found. Bottom right: False detections eliminated.

positives we were suffering from with the classifier alone are eliminated. The focus of attention is calculated in ca 1.5 sec. and the classification at these region of interest needs 200 ms (image size: 240 × 320, Pentium-M 1.7 GHz). The bulk of the running time of VOCUS is taken up by the feature computations. These may be parallelized by splitting up the processing to several CPUs [9] or with dedicated hardware [18] what makes the system real-time capable. We consider this for future work.

V. CONCLUSIONS

Using the visual attention system VOCUS combined with a fast classifier, we have designed a robust ball detection system with a very low misclassification rate, even in complex, cluttered images. Due to the use of an edge detection Sobel filter and a threshold to preprocess the training images for the cascade, the classifier is color-invariant, leaving the color to be learned by the attention system. Assuming short-term prior knowledge about the ball to be used for a RoboCup match, VOCUS is quickly adjusted to the ball with very few images.

The success of the algorithm is reached by only searching for balls in regions hypothesized by the attention algorithm to contain the ball, thereby eliminating false positives. Although the algorithm misses a few balls, what we are concerned with is how it will perform in the RoboCup environment. In this case, the reliability of the algorithm seems to be sufficient. Even if the ball is not detected in one in every 5 pictures, for example, the robot will still be able to follow it quite confidently.

Needless to say, much work remains to be done: As the detection of regions of interest is currently relatively slow compared to the ball detection, the next step is to work on increasing the efficiency of the attention system and therefore of the whole detection scheme. In addition it is planned to enhance the presented algorithms by adding time dependent behavior either by using standard tracking with particle filters or by using a time dependent attention control.

REFERENCES

[1] G. Backer, B. Mertsching, and M. Bollmann. Data- and model-driven gaze control for an active-vision system. *IEEE Trans. on Pattern Analysis & Machine Intelligence*, 23(12):1415–1429, 2001.

[2] R. E. Burger. *Colormanagement. Konzepte, Begriffe Systeme*. Springer, 1997.

[3] M. Oren C. Papageorgiou and T. Poggio. A general framework for object detection. In *Proceedings of the 6th International Conference on Computer Vision (ICCV '98)*, Bombay, India, January 1998.

[4] M. Das and J. Anand. Robust Edge detection in noisy images using and adaptive stochastic gradient technique. In *Proc. ICIP*, 1995.

[5] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *Machine Learning: Proc. of the 13th Int. Conf.*, 1996.

[6] S. Frintrop, G. Backer, and E. Rome. Goal-directed search with a top-down modulated computational attention system. *submitted*.

[7] S. Frintrop, E. Rome, A. Nüchter, and H. Surmann. A bimodal laser-based attention system. *CVIU*, (ccepted).

[8] RoboCup. `http://www.robocup.org`.

[9] L. Itti. Real-time high-performance attention focusing in outdoors color video streams. In *SPIE Human Vision and El. Im. IV*, 2002.

[10] T. Balch J. Bruce and M. Veloso. Fast and inexpensive color image segmentation for interactive robots. In *Proc. IROS*, 2000.

[11] C. Koch and S. Ullman. Shifts in selective visual attention: towards the underlying neural circuitry. *Human Neurobiology*, 1985.

[12] C. Koch L. Itti and E. Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE PAMI*, 20(11), 1998.

[13] R. Lienhart and J. Maydt. An Extended Set of Haar-like Features for Rapid Object Detection. In *Proc. of the IEEE Conf. on Image Processing (ICIP '02)*, pages 155 – 162, New York, USA, 2002.

[14] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International J. of Computer Vision*, 60(2):91–110, 2004.

[15] F. Miau, C. Papageorgiou, and L. Itti. Neuromorphic algorithms for computer vision and attention. In *Proc. SPIE 46 Ann. Int. Symp. on Optical Science and Technology*, vol. 4479, Nov. 2001.

[16] S. Mitri, K. Pervölz, A. Nüchter, and H. Surmann. Fast Color-Independent Ball Detection for Mobile Autonomous Robots. In *Proc. IEEE Mechrob*, Aachen, Germany, 2004.

[17] U. Neisser. *Cognitive Psychology*. Appleton-Century-Crofts, 1967.

[18] N. Ouerhani and H. Hügli. Real time visual attention on a massively parallel simd architecture. *J. of Real-time imaging*, 9(3), 2003.

[19] R. Haneka T. Bandlow, M. Klupsch and T. Schmitt. Fast image segmentation, object recognition and localization in a robocup scenario. In *3. RoboCup Workshop, IJCAI*, 1999.

[20] A. Treptow and A. Zell. Real-time object tracking for soccer-robots without color information. *Journal Robotics and Autonomous Systems*, 48(1):41 – 48, August 2004.

[21] John K. Tsotsos. Complexity, vision, and attention. In *Vision and Attention*, chapter 6. 2001.

[22] P. Viola and M. J. Jones. Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154, May 2004.

[23] D. Walther, U. Rutishauser, Chr. Koch, and P. Perona. On the usefulness of attention for object recognition. In *Proc. WAPCV*, 2004.

ACKNOWLEDGMENTS

We would like to thank G. Backer, M. Hennig, J. Hertzberg, and E. Rome for supporting our work.